VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Timo Huhtakallio

# WRM247:N ETÄPÄIVITYS

Tekniikka ja liikenne
2012

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

# TIIVISTELMÄ

| | |
|---|---|
| Tekijä | Timo Huhtakallio |
| Opinnäytetyön nimi | WRM247:n etäpäivitys |
| Vuosi | 2012 |
| Kieli | suomi |
| Sivumäärä | 23 |
| Ohjaaja | Kalevi Ylinen |

Työn tarkoituksena oli suunnitella ja toteuttaa etäpäivitysohjelmakirjasto Wapice Oy:n WRM247-laitteelle, joka on suunniteltu yhdistämään elektroniset laitteet ympäri maailman WRM-järjestelmään.

Ohjelmakirjasto on tarkoitettu laitteen pääohjelman käytettäväksi aloittamaan päivitys, kun uusi laiteohjelmaversio, firmware, on saatavilla. Tuotannossa laiteohjelma julkaistaan WRM-palvelimella, joka myös käynnistää päivityksen.

WRM-järjestelmä on vielä sisäisessä kehityksessä, joten kirjasto testattiin käyttäen testiohjelmaa. Testiohjelma käytti kirjastoa ja WRM:n DDCT-työkalua (Device Discovery and Configuration Tool), jolla päivitys saatiin käynnistettyä yrityksen sisäisessä lähiverkossa käyttäen TCP/IP-verkon broadcast-viestejä lähettämään komentoja tietokoneen ja sulautetun laitteen välillä.

Työssä toteutettiin ainoastaan sisäverkossa toimiva versio. Lopulliseen versioon tullaan tarvitsemaan myös delta-päivitysominaisuus hitaita verkkoja varten

Avainsanat etäpäivitys, sulautettujärjestelmä, linux, WRM247

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

# ABSTRACT

| | |
|---|---|
| Author | Timo Huhtakallio |
| Title | Remote Update of WRM247 |
| Year | 2012 |
| Language | Finnish |
| Pages | 23 |
| Name of Supervisor | Kalevi Ylinen |

The purpose of this thesis was to design and implement remote update software library for Wapice's WRM247, remote management system, which is meant to be used worldwide to connect various electrical systems to WRM service.

Software library is used by the devices, main application to launch the update when the new firmware version is available. In production, the new firmware will be published on WRM server that triggers also update process.

The whole WRM system was still in internal development, so the library was tested by using test application that uses library and WRM's DDCT, Device Discovery and Configuration Tool, to enable possibility to start update in company's internal network using TCP/IP networks broadcast messages to transfer commands between PC and embedded device.

Within the scope of the thesis study only an internal network version was published. For further development, a delta update possibility will need to be implemented for slow networks.

Keywords Remote update, embedded systems, linux, WRM247

# MERKINNÄT JA LYHENTEET

| | |
|---|---|
| ARM | Advanced RISC Machines. Sulautetuissa järjestelmissä laajasti käytössä oleva prosessoriarkkitehtuuri. |
| CAN | *Controller-Area Network.* Teollisuudessa yleisessä käytössä oleva sarjaliikennekommunikointijärjestelmä |
| Eclipse | Avoimen lähdekoodin lisenssillä toimiva kehitysympäristö. |
| FTP | File Transfer Protocol, Tiedostojen siirtoprotokolla. |
| GCC | *GNU Compiler Collection.* Avoimen lähdekoodin lisenssillä toimiva kääntäjäkokoelma. |
| GNU | GNU's Not Unix. Järjestö joka on omistautunut vapaan lähdekoodin ohjelmistoihin. |
| GPRS | General Packet Radio Service. GSM-verkossa toimiva pakettikytkentäinen tiedonsiirtopalvelu. |
| JFFS2 | Journaling Flash File System version 2. Flash muisteille suunniteltu tiedostojärjestelmä. |
| SMA | SubMiniature version A. Pääasiassa koaksiaalikaapeleiden kytkentöihin käytetty liitintyyppi. |
| TCP/IP | Transmission Control Protocol / Internet Protocol, internetin tiedonsiirtoprotokollakokokoelma. |
| UBIFS | Unsorted Block Image File System. JFFS2:n seuraaja. |
| VPN | Virtual Private Network, virtuaalinen lähiverkko |

# SISÄLLYS

# 1 JOHDANTO

## 1.1 Wapice Oy

Wapice Oy on vaasalainen vuonna 1999 perustettu työntekijöiden omistama teollisuuteen suuntautunut ohjelmistoyritys, joka työllistää noin 200 henkilöä. Yrityksen pääkonttori sijaitsee Vaasassa ja sillä on haarakonttoreita Vaasan Runsorissa, Oulussa, Seinäjoella, Tampereella ja Hyvinkäällä. /1/

Wapicen toiminta jakautuu kolmeen segmenttiin: Sulautetut järjestelmät, Teollisuusjärjestelmät ja Liiketoimintaratkaisut. /2/

Ohjelmistoalihankinnan lisäksi yrityksellä on omina tuotteina Summium-myyntikonfiguraattori ja kehitysvaiheessa oleva WRM-etähallintajärjestelmä. /3/

## 1.2 Työn sisältö

Työn tarkoituksena oli suunnitella ja toteuttaa Wapice Oy:n WRM-etähallintajärjestelmälle, Wapice Remote Management, WRM247-laitteen etäpäivitysratkaisu. ARM9-prosessoriarkkitehtuuriin perustuvan linux-laitteen oletettu elinikä on yli 15 vuotta, jonka seurauksena laitteen käyttöjärjestelmä ja ohjelmisto joudutaan todennäköisimmin päivittämään kesken sen elinkaaren.

Laitteen käyttökenttää ei ole rajattu maantieteellisesti, joten päivityksen on tapahduttava ilman, että ylläpitäjän täytyy mennä paikallisesti päivittämään laite, tai että laitetta tarvitsee lähettää valmistajalle päivitettäväksi.

Opinnäytetyössä esitetty päivitysohjelma toteutettiin linux-kirjastoksi laitteen pääohjelman käytettäväksi. Työn toteutusaikana laitteen ohjelmisto oli vielä yrityksen sisäisessä kehityksessä ja pilotointivaiheessa.

Kirjaston toimivuus varmistettiin erillisellä testiohjelmalla, joka kommunikoi PC-tietokoneen kanssa oman protokollan kautta, mikä käytti TCP/IP-verkon broadcast-viestejä.

# 2 WRM-ETÄHALLINTAJÄRJESTELMÄ

WRM-etähallintajärjestelmä, Wapice Remote Management, on Wapicen suunnittelema asiakkaiden tarpeiden mukaan räätälöitävä palvelu, joka käsittää WRM247 tai WRM365 sulautetun linux-laitteen ja internet-palvelimen, johon laitteet ovat jatkuvasti tai periodisesti yhteydessä.

## 2.1 WRM247

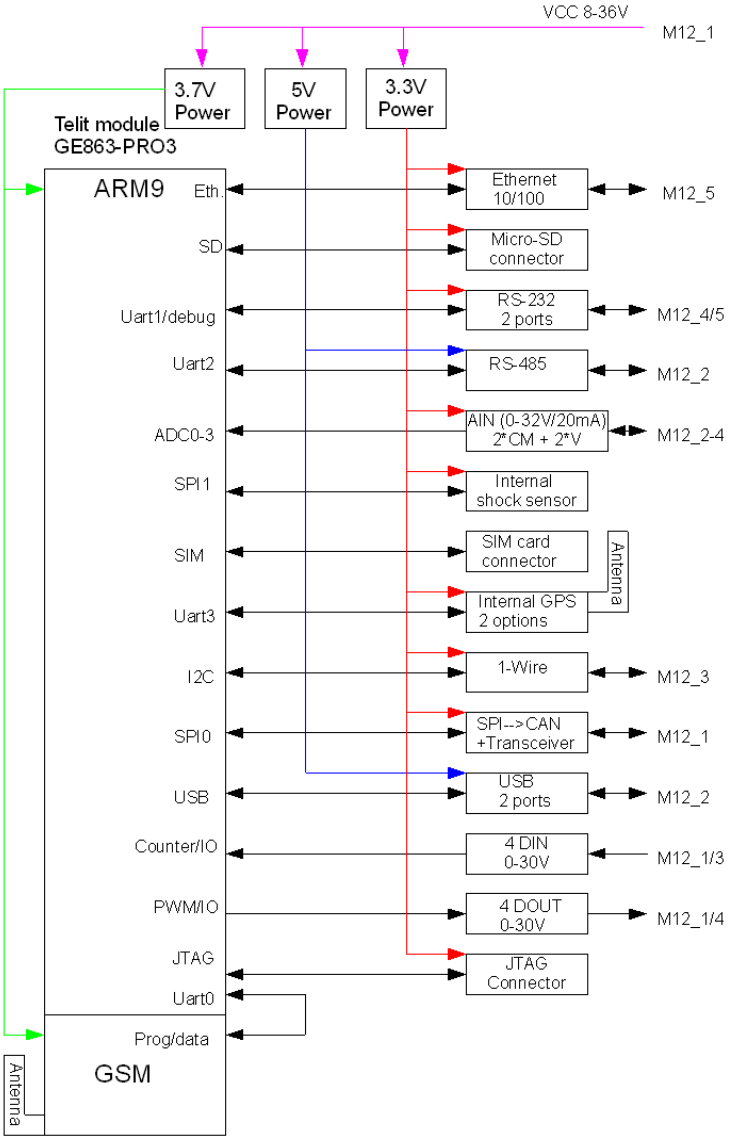
WRM247 on Wapice Oy:n kehittämä etähallintalaite, jonka avulla lähes mikä tahansa elektroninen laite saadaan liitettyä WRM-järjestelmään tarkkailtavaksi ja/tai ohjattavaksi. Laitetta voidaan käyttää myös yksistään ilman WRM-palvelinta.


Kuva 1. WRM247-laitteen piirilevy

Laitteen ytimenä toimii Telitin valmistama GE863Pro3 200 MHz:n ARM9-GSM-moduuli, kuvassa 1 alhaalla oikealla. Laitteessa on viisi M12-liitintä erilaisia kytkentärajapintoja varten. Laitteessa on neljä kappaletta digitaalisia sisääntuloja ja -ulostuloja, neljä analogista (0 - 32 V) sisääntuloa, joista kaksi on konfiguroitavissa 0-22 mA virtaviestiksi, ethernet, RS232 ja RS485, kaksi CAN-

väylää ja 1-wire. Sekä GPS:n ja GSM:n antennit SMA- liittimissä. Lisäksi laitteessa on sisäänrakennettu kiihtyvyysanturi ja microSD-muistikorttipaikka. /4/

Kuvassa 2 on lohkokaavio laitteen rakenteesta. Kuvasta näkyy, mihin rajapinnat on prosessorilla kytketty ja mistä liittimestä ne tulevat ulos laitteelta.




Kuva 2. WRM247-laitteen lohkokaavio

## 2.2 Telit

Telit on italialainen vuonna 1986 perustettu alun perin tutkimus- ja kehityspalveluita myynyt telekommunikaatioyritys. Vuonna 1998 yritys julkaisi ensimmäisen M2M-moduulinsa nimeltä Datablock. Nykyään yritys valmistaa useita erilaisia prosessori- ja radiomoduuleja, joita on markkinoilla yli 80 maassa. /5/

## 2.3 ARM

ARM Holdings on maailman johtava puolijohdeimmateriaalioikeuksien tuottaja. Sen pääkonttori sijaitsee Cambridgessa Iso-Britanniassa ja se työllistää yli 2000 henkilöä. Toimistoja ja suunnittelukeskuksia ARMilla on Taiwanissa, Ranskassa, Intiassa, Ruotsissa ja Yhdysvalloissa. ARM-teknologiaa käytetään 90 % älypuhelimissa, 80 % digitaalikameroissa ja 28 % elektronisissa laitteissa. /6/

## 2.4 Protokollat

Työssä käytettyjä keskeisimpiä protokollia ovat verkkoprotokollat TCP/IP ja sen alle sijoittuva FTP.

TCP/IP on laaja kokoelma erilaisia verkkoprotokollia, jotka perustuvat alun perin TCP- ja IP- protokolliin. Kaikki internetin liikenne perustuu TCP- ja IP-protokolliin.

IP, Internet Protocol, pitää huolen verkossa olevien koneiden välisestä tiedonsiirrosta. Jokaisella internetissä olevalla tietokoneella on oma uniikki IP-osoite, jonka perusteella siihen saadaan yhteys.

TCP:ta, Transmission Control Protocol, käytetään verkossa tiedon siirtämiseen kahden ohjelman välillä. TCP on vastuussa tiedostojen pilkkomisesta IP-paketteihin ennen kuin ne lähetetään ja niiden kokoamiseen vastaanottopäässä.

FTP, File Transfer Protocol, on TCP-protokollaa käyttävä tiedostojen siirtoprotokolla. FTP-protokollassa asiakas, client, muodostaa yhteyden

palvelimeen, host/server. Palvelin voi olla julkinen, johon kuka tahansa voi ottaa yhteyden, tai käyttäjätunnuksella ja salasanalla suojattu.

# 3 TOTEUTUS

## 3.1 Kehitysympäristö

Kehitysympäristönä työssä oli yleisesti WRM-kehityksessä käytetty Oracle VM VirtualBox:iin asennettu Ubuntu 10.04 LTS –linux-käyttöjärjestelmä. Asentamalla kehitysympäristö virtuaalikoneeseen saadaan helposti pystytettävä ja siirreltävä aloituspaketti, esimerkiksi uudelle kehittäjälle tai uuteen tietokoneeseen. Useimmissa tietokoneissa on Windows-käyttöjärjestelmä /7/ ja kahden rinnakkaisen käyttöjärjestelmän käyttäminen on hankalampaa ja hitaampaa, kuin toisen käyttöjärjestelmän virtualisointi.

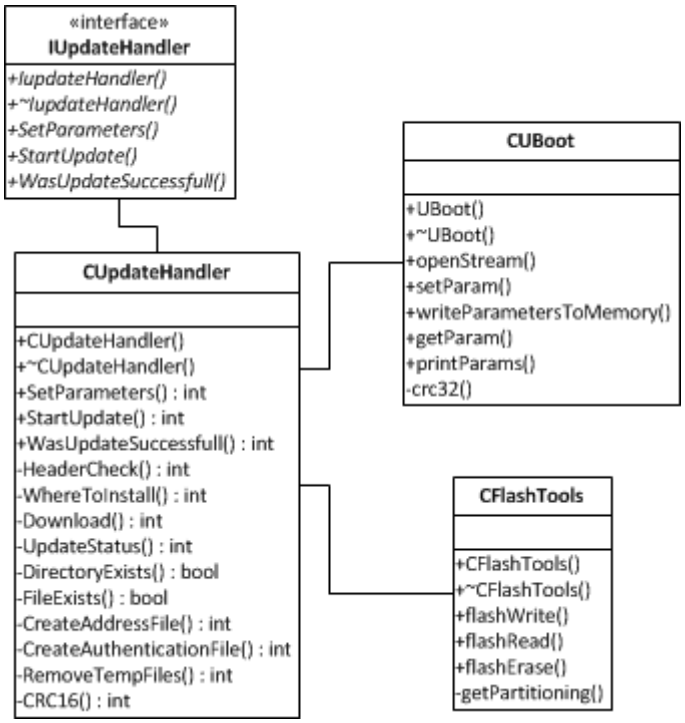
Ohjelmointiympäristönä käytettiin Eclipseä, koska se on opensource eli avoimen lähdekoodin ohjelma ja toimii niin linuxissa kuin Windowsissakin. Eclipse on IBM:n vuonna 2001 aloittama projekti, joka muuttui vuonna 2004 itsenäiseksi Eclipse Foundationiksi /8/. Eclipseä käytettiin myös siksi, että siihen on saatavilla SVN plug-in, subclipse, jolla koodit saadaan synkronoitua helposti suoraan ohjelmointiympäristöstä yrityksen versiohallintaan ilman erillistä ohjelmaa, tai käyttämällä linuxin komentoriviä.

Ubuntuun otettiin Telitin omasta coLinux-kehitysympäristöstä ladattu arm-linux-uclibc-gcc -ristikääntäjä, ristikääntäjän avulla PC-tietokoneella tuotettu koodi saadaan käännettyä kohdelaitteella suoritettavaksi. Telitin tarjoamaa kehitysympäristöä ei käytetty, koska se toimi vain 32-bittisissä käyttöjärjestelmissä.

## 3.2 Ohjelmointi

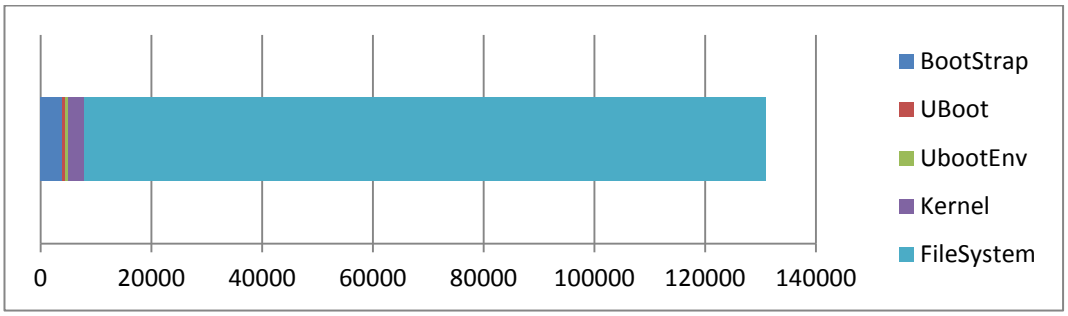
Kirjasto toteutettiin C++ -ohjelmointikielellä, jolloin siihen saatiin luotua selkeä luokkarakenne, jolla saadaan rajattua erityyppiset toiminnallisuudet omiin luokkiinsa. Kirjaston rajapinta pyrittiin saamaan mahdollisimman yksinkertaiseksi ja se onnistuttiin tiivistämään kolmeen metodiin, joista yksi on polymorphinen, eli sitä voidaan kutsua käyttäen erilaisia parametriyhdistelmiä. Rajapinnan alla on

monimutkaisempi luokkarakenne, jonka toiminnallisuuksista ja suhteista käyttäjän ei tarvitse huolehtia. Kuvassa 3 on luokkakaavio kirjaston rakenteesta.

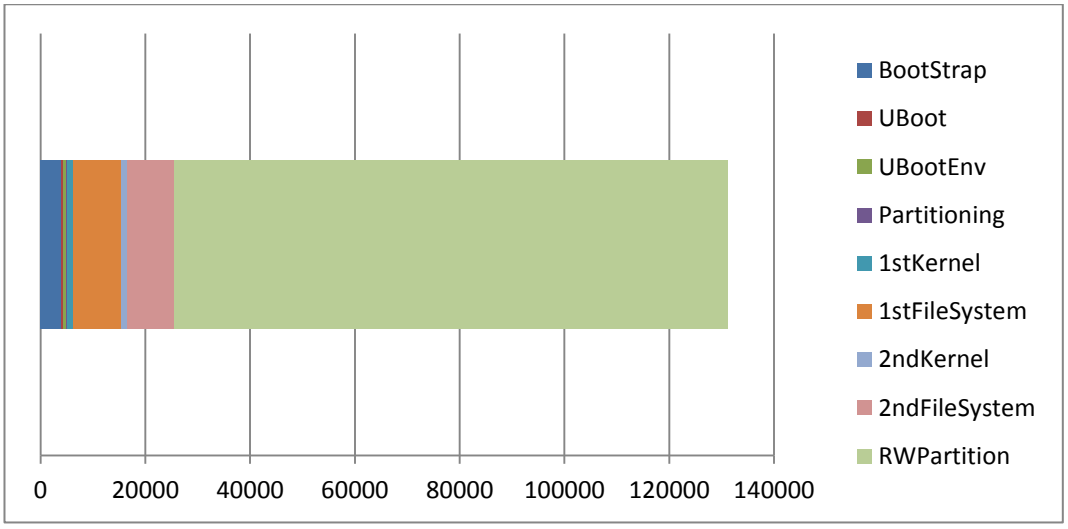



Kuva 3. Luokkakaavio

Laitteen alkuperäiseen tiedostojärjestelmän rakenteeseen, kuva 4, verrattuna päivitystä varten toteutettiin täysin erilainen osiointi, kuva 5, jossa on osiot kahdelle rinnakkaiselle linuxin ytimelle eli kernelille ja tiedostojärjestelmälle. Tiedostojärjestelmään lisättiin myös staattinen osio taulukolle, jossa määritellään osioiden alkuosoitteet ja koot, jos osiointia halutaan myöhemmin muuttaa. Lisäksi käytössä oleva ohjelmisto-osio muutettiin vain luku -tilaan datan korruptoitumisen välttämiseksi ja lisättiin osio, johon käytön aikana voidaan kirjoittaa.




Kuva 4. Laitteen alkuperäinen osiointi




Kuva 5. Muokattu osiointi

Laitteen U-Bootiin jouduttiin toteuttamaan logiikka, jolla voidaan ajaa linuxia kahdesta sijainnista riippuen, kummassako osiossa käytettäväksi merkitty ohjelmisto sijaitsee. Kun ensimmäisellä osiolla sijaitseva käyttöjärjestelmä on käytössä, uusi ohjelmistopäivitys kirjoitetaan jälkimmäiselle osiolle. Päivityksen jälkeen ohjelma käynnistää laitteen uudelleen ja aloittaa uuden käyttöjärjestelmän käytön jälkimmäiseltä osiolta. Seuraavassa päivityksessä uusi ohjelmisto kirjoitetaan ensimmäiseen osioon.

### 3.3 IUpdateHandler

IUpdateHandler on ohjelmointirajapinta kirjastolle. Tämän luokan funktioita voidaan kutsua muualta ja kutsujan ei tarvitse välittää niiden takana tapahtuvista prosesseista.

### 3.3.1 Virtual SetParameters()

Virtual SetParameters() -metodi ottaa vastaan ja tallentaa päivitykseen tarvittavat tiedot. Metodi on polymorphinen eli sitä voidaan kutsua erilaisilla parametreilla. Tässä tapauksessa sallitaan kutsuminen merkkijonolla, joka sisältää uuden ohjelmistopäivityksen sijainnin sekä lisäksi kahdella valinnaisella merkkijonolla joka sisältää palvelimen käyttäjätunnuksen ja salasanan, jos pääsyyn vaaditaan todennus. Toiminnon onnistuessa metodi palauttaa OK:n ja virhetilanteessa vikakoodin.

### 3.3.2 Virtual StartUpdate()

Virtual StartUpdate() -metodin alla tapahtuu ohjelmistopäivityksen lataaminen, tarkastukset ja levylle kirjoittaminen.

### 3.3.3 Virtual WasUpdateSuccessfull()

Virtual WasUpdateSuccessfull() -metodi ajetaan laitteen uudelleenkäynnistyksen yhteydessä, kun uutta ohjelmistopäivitystä otetaan ensimmäistä kertaa käyttöön. Metodi tarkastaa, että laite toimii konfiguraation mukaisesti ennen kuin se vahvistaa, että uusi ohjelmisto merkitään jatkossa käytettäväksi. Jos metodi epäonnistuu päivityksen jälkeen kolmen käynnistyksen yhteydessä, se palauttaa edellisen ohjelmistoversion käyttöön.

## 3.4 CUpdateHandler

CUpdateHandler-luokka periytyy IUpdateHandler-luokasta ja toteuttaa täten pääluokan metodit sekä joukon omia yksityisiä metodeita, jotka eivät ole käyttäjälle näkyviä.

### 3.4.1 HeaderCheck()

HeaderCheck() -metodi lukee ladatun ohjelmistopaketin alusta uuden ohjelmiston version päivitytarpeen ja datasisällön tarkastussumman tiedoston eheyden varmistamiseksi.

### 3.4.2 WhereToInstall()

WhereToInstall() -metodi lukee linuxin osioinnista, kumpi järjestelmistä on kaytössä ja palauttaa vapaan osion osoitteet.

### 3.4.3 Download()

Download() -metodi aloittaa ohjelmiston lataamisen palvelimelta hakemiensa parametrien, URL-osoitteen ja mahdollisesti käyttäjätunnuksen ja salasanan, perusteella. Paketti haetaan palvelimelta linuxin *wget*-ohjelman avulla, joka on myös GNU-projektin ohjelma. Tämän sulautetun linuxin rajoitteiden takia latausosoite jouduttiin parsimaan yksinkertaistettuun muotoon, jonka jälkeen komennoksi muodostuu wget –c ftp://user:pass@server.org/fw.tar.gz. Normaalisti käyttäjätunnus ja salasana asetetaan komennon perään parametreillä --user ja --password /9/.

Parametri –c, continue, asetetaan metodille annetun boolean -tyyppisen parametrin perusteella, joka määrittelee jatketaanko latausta, jos tallennusosoitteessa on jo osittainen päivityspaketti. Tilanne voi ilmetä laitteen uudelleenkäynnistymisen tai verkkoyhteyden menettämisen seurauksena kesken edellisen päivitysprosessin.

### 3.4.4 UpdateStatus()

Päivitysprosessin eri vaiheissa tallennetaan muuttujaan prosessin sen hetkinen tila. Tämän metodin avulla luetaan turvallisesti prosessin tila, ja sen perusteella voidaan tehdä jatkotoimia. Jatkotoimia tarvitaan, esimerkiksi jos lataus on katkennut ja paketti on viallinen, niin ettei sen latausta voida jatkaa. Paluuarvon

perusteella voidaan keskeneräinen tiedosto määrätä poistettavaksi ja aloittaa lataus uudelleen.

### 3.4.5 DirectoryExists()

Ennen latauksen aloittamista täytyy varmistaa, että sijainti, johon tiedosto aiotaan tallentaa, on olemassa, sillä linuxin wget-lataus epäonnistuu, jos kohdekansiota ei ole olemassa. Metodissa avataan dirent-standardikirjaston funktiolla opendir haluttu kansio. Jos funktio onnistuu, se palauttaa 0, voidaan todeta, että kansio on olemassa ja tämän jälkeen kansio suljetaan closedir-funktiolla ja vapautetaan se muiden prosessien käyttöön.

```
bool CUpdateHandler::DirectoryExists(const std::string& dirname){

	DIR* dir;
	if((dir = opendir(dirname.c_str())))
	{
		closedir(dir);
		return true;
	}
	return false;
}
```

### 3.4.6 FileExists()

FileExists() on samankaltainen metodi kuin DirectoryExists(), mutta tarkastetaan tiedoston olemassaolo yrittämällä avata tiedosto stream-metodin konstruktorissa. Jos konstruktorin suoritus onnistuu, todetaan tiedoston olemassaolo ja suljetaan tiedosto.

### 3.4.7 CreateAddressFile()

CreateAddressFile() -metodissa tallennetaan palvelimelta saatu päivitysosoite tiedostoon, josta voidaan odottamattoman yhteyden katkeamisen tai uudelleenkäynnistyksen jälkeen tarkastaa päivityksen jatkamisen tarve ilman palvelimen uutta kehotetta.

### 3.4.8 CreateAuthenticationFile()

CreateAuthenticationFile() -metodi tallentaa palvelimen käyttäjätunnuksen ja salasanan samaa käyttöä varten kuin CreateAddressFile() päivitysosoitteen.

### 3.4.9 RemoveTempFiles()

RemoveTempFiles() -metodi poistaa päivityksen jälkeen onnistuneen tai epäonnistuneen päivityksen aikana luodut ja ladatut tiedostot laitteen rajallisen tilan säästämiseksi.

### 3.4.10 CRC16()

CRC16() -metodin avulla tarkastetaan päivityspaketin datan eheys. Paketin ”headeriin” on tallennettu ennen siirtoa laskettu CRC16-tarkistussumma. CRC16 on kaksi tavua, 16 bittiä pitkä datasta laskettu CRC (Cyclic Redundancy Check) tarkistussumma, jonka avulla tarkastetaan data virheiden varalta. Tekniikan avulla voidaan vain havaita virheet, ei korjata niitä. /10/

Metodille annettiin parametreina osoitin dataan ja data-alueen pituus. Metodi palauttaa tarkistussumman annetulle datalle. Alla on koodi, jolla tarkistussumma laskettiin:

```
unsigned int CUpdateHandler::crc16(char* data_p, unsigned int length){

	unsigned int crc = 0;

	for (unsigned int j = 0; j < length; j++)
	{
		unsigned int b = data_p[j];
		for (unsigned int i = 0; i < 8; i++)
		{
			crc = ((b ^ (unsigned int)crc) & 1) ? ((crc >> 1) ^
			0xA001) : (crc >> 1);
			b >>= 1;
		}
	}
	return crc;
}
```

## 3.5 UBoot

### 3.5.1 openStream()

OpenStream() -metodi avaa U-Bootin parametriosion stream-tyyppiseen standardimerkkijonoon ja parsii merkkijonosta tyyppi-arvo pareja.

### 3.5.2 setParam()

SetParam() -metodi lisää uuden tai muokkaa olemassa olevaa U-Boot parametria. Jos parametri on olemassa, sen arvo korvataan uudella ja parametrin puuttuessa se luodaan ja sille määritellään annettu arvo. Metodi ainoastaan tallettaa muutokset ohjelman parametreihin, jotka katoavat uudelleenkäynnistyksessä.

### 3.5.3 writeParametersToMemory()

Kun kaikki muokattavat parametrit on talletettu, WriteParametersToMemory() -metodi kirjoittaa ne laitteen pysyvään flash-muistiin yhdellä kertaa, kirjoituskertojen vähentämiseksi.

### 3.5.4 getParam()

GetParam() -metodi palauttaa pyydetyn parametrin arvon, jos parametria ei ole olemassa, virhekoodi tallennetaan annettuun muuttujaan.

### 3.5.5 printParams()

PrintParams() tulostaa U-Bootin parametrien nimet ja arvot. Tuotannossa metodilla ei ole käyttöä, mutta kehitysvaiheessa oli päivityksen yhteydessä hyvä nähdä miten parametrien luku ja kirjoitus onnistui.

### 3.5.6 crc32()

Crc32() on samankaltainen metodi kuin CUpdateHandlerin crc16(), mutta koska U-Boot käyttää tätä tarkistussummaa parametrien eheyden varmistamiseen,

tarkistussumman pituus on neljä tavua. Ympäristöparametriosiolla tämä summa kirjoitetaan osion alkuun ennen itse parametreja.

## 3.6 FlashTools

### 3.6.1 flashWrite()

FlashWrite() -metodi kirjoittaa puskurissa olevan datan levylle parametrin osoittamaan paikkaan. Metodi toteutettiin aluksi normaalin tiedostokirjoitusfunktion avulla, mutta sen käyttö jouduttiin korvaamaan käyttöön sopivaksi muokatulla, linuxin omalla mtd-util/nandwrite-funktiolla. Tämä siltä varalta, että alueella, johon päivitystä kirjoitetaan, olisi muistissa viallinen sektori eli badblock. Linuxin nandwrite:ssä on logiikka viallisten sektorien ohittamiseen kirjoitusvaiheessa. U-Bootin lataustyökalu ohittaa vialliset sektorit tarkistaessaan ohjelmiston yhtenäisyyttä.

### 3.6.2 flashRead()

FlashRead() lukee halutun määrän tavuja alkaen osion alusta lähtien. Metodille ei tässä toteutuksessa ollut käyttöä, mutta sen yksinkertaisuudesta johtuen se toteutettiin vastaisuuden varalle.

### 3.6.3 flashErase()

FlashErase() -metodi tyhjentää halutun osion muistista ja alustaa sen annetun parametrin mukaisesti joksikin linuxin tiedostojärjestelmäksi, esimerkiksi JFFS2 tai UBIFS.

### 3.6.4 getPartitioning()

GetPartitioning() lukee linuxin /proc/mtd-tiedostosta järjestelmän osiointitiedon ja tallentaa osioiden tiedot struct:iin, johon kerätään mtdblockin enumeraatio, koko ja nimi. Nimen alun kahdesta merkistä voidaan lukea osion tyyppi. Tyyppi määrittelee, mikä osa ohjelmistosta osiolle tullaan tallettamaan.

```
typedef struct{
	unsigned char block;
	unsigned char code;
	unsigned int size;
	char name[20];

}Partition;
```

# 4 TESTAUS

Kirjaston toimivuus todettiin suorittamalla sillä erilaisia suoritustilanteita. Testitapaukset tallennettiin TestLink-testauksen hallintatyökaluun, johon saadaan tallennettua myös tulokset testien lopputuloksista ja tarvittaessa muodostettua raportti testeistä. Kirjasto testattiin seuraavilla tilanteilla, suluissa hyväksyttyyn suoritukseen oikeuttava lopputulos:

1. Päivitä käyttäen ftp-palvelinta (onnistui)
2. Päivitä käyttäen http-palvelinta (onnistui)
3. Yritä päivittää väärällä käyttäjätunnuksella (virhe)
4. Yritä päivittää väärällä salasanalla (virhe)
5. Yritä päivittää väärästä osoitteesta (virhe)
6. Simuloi virtakatkos kesken latauksen (palautui)
7. Simuloi virtakatkos kesken muistiinkirjoituksen (palautui)
8. Päivitä toimimattomalla ohjelmistopäivityksellä (palautui)
9. Päivitä vahingoittuneella ohjelmistopäivityksellä (virhe)

Virhetilanteissa hyväksyntään vaadittiin, että laitteen toiminta jatkuu normaalisti päivitysyrityksen jälkeen.

# 5 YHTEENVETO JA TULOKSET

Kirjaston testauksessa varmistettiin, että päivitys onnistuu toimivalla ohjelmistopäivityksellä molemmille osioille ja, että virhetilanteessa laitteen toiminta palautuu normaaliksi aiheuttamatta merkittäviä häiriöitä laitteen normaalille toiminnalle. Sallituksi häiriöksi luettiin laitteen toiminnan tai kommunikaation hetkellinen hidastuminen ja yhteyden hetkellinen katkeaminen tilanteessa, jossa uusi ohjelmistopäivitys ei ollut toimiva.

Kirjaston käyttö helpotti kehittäjien työtä, sillä kirjaston ja siihen toteutettujen testiohjelmien avulla laite pystyttiin päivittämään muutamalla napin painalluksella ja kehittäjän ei tarvinnut enää olla samassa paikassa laitteen kanssa.

Ongelmia testivaiheessa aiheutti ratkaisun puuttuminen GSM-verkon yli tapahtuvan päivityksen käynnistämisessä ja, että komentoihin käytetyt broadcast-viestit eivät kulkeneet VPN-yhteyden yli.

Jatkokehityksen kannalta usean megatavun kokoisen ohjelmiston päivittämisen rinnalle olisi hyvä saada niin sanottu delta-päivitysmahdollisuus, jossa ohjelmistossa päivitetään vain versioiden välillä muuttuneet osat, jolloin päivitystä saadaan nopeutettua hitaissa verkoissa ja siirrettävän datan määrää pienennettyä mahdollisessa roaming-tilassa.

# LÄHTEET

/1/ http://www.wapice.com/wapice_cms/fi/news/165-wapices-net-sales-grew-41-in-first-quarter-2012.html Viitattu 8.5.2012.

/2/ http://www.wapice.com/wapice_cms/fi/ratkaisut.html Viitattu 8.5.2012.

/3/ http://www.wapice.com/wapice_cms/fi/paeaesivu.html Viitattu 8.5.2012.

/4/ http://www.wrm.fi/files/WRM_247_rgb_full.pdf Viitattu 8.5.2012.

/5/ http://www.telit.com/en/about/company.php Viitattu 8.5.2012.

/6/ http://arm.com/about/company-profile/index.php Viitattu 8.5.2012.

/7/ http://netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=0 Viitattu 22.5.2012.

/8/ http://www.eclipse.org/org/#history Viitattu 22.5.2012.

/9/ http://www.gnu.org/software/wget/manual/wget.html Viitattu 26.5.2012.

/10/ http://www.hackersdelight.org/crc.pdf Viitattu 27.5.2012.